Package 'phdid'

September 15, 2026

Title Partial Homogeneity in Staggered Difference-in-Differences

Version 0.1.0

Description In staggered difference-in-differences designs the treatment effect is a vector of cohort-time effects rather than a single number. Estimating each separately is unbiased but imprecise when some are equal, while pooling them all is precise but biased under genuine heterogeneity. This package treats the choice as a partition-selection problem on the cohort-time cells and provides two estimators for it: a Dirichlet process mixture fitted by a collapsed Gibbs sampler, whose posterior marginalises over the unknown partition and reports co-clustering probabilities, and an 'L0'-penalised estimator that returns a single partition and arises as the fixed-variance maximum a posteriori solution of the same model. Also provides tests for whether the cohort-time effects carry recoverable heterogeneity at all, sampler diagnostics including exact enumeration of the partition posterior for small designs, regularisation paths for both estimators, and a calibrated data-generating process. All estimators accept a vector of first-stage cohort-time effects with their joint covariance, so any heterogeneity-robust first-stage estimator may be used. Methods are described in Arora and Wagle (2026) <doi:10.2139/ssrn.7207083>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports graphics, grDevices, stats, utils

Suggests did (>= 2.1.0), knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr, rmarkdown

URL https://github.com/ronwag2005/phdid

BugReports https://github.com/ronwag2005/phdid/issues

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Rohan Wagle [aut, cre, cph],
Parush Arora [aut, cph]

Maintainer Rohan Wagle <rohan.wagle_ug2023@ashoka.edu.in>

Repository CRAN

Date/Publication 2026-09-15 12:00:02 UTC

Contents

`adjusted_rand` *Adjusted Rand index between two partitions*

Description

Used to score recovery of the true partition in the simulations (paper Tables 2 and 3).

Usage

```
adjusted_rand(a, b)
```

Arguments

a, b	integer vectors of group labels of equal length.

Value

the adjusted Rand index; 1 for identical partitions, 0 in expectation for independent ones.

Examples

```
adjusted_rand(c(1, 1, 2, 2), c(2, 2, 1, 1))  # same partition, relabelled
adjusted_rand(c(1, 1, 2, 2), c(1, 2, 1, 2))
```

---

aggregate.bayes_ph *Aggregate cohort-time effects into a reported estimand*

Description

The cohort-time effects are rarely the final object; papers report an overall ATT, an event-study profile, or effects by cohort. All of these are linear aggregators $\sum_k w_k \tau_k$ of the cell effects, so the specification question is orthogonal to the choice of aggregator.

Usage

```
## S3 method for class 'bayes_ph'
aggregate(
  x,
  type = c("overall", "dynamic", "group", "calendar"),
  weights = NULL,
  level = 0.95,
  ...
)

## S3 method for class 'ph_fit'
aggregate(
  x,
  type = c("overall", "dynamic", "group", "calendar"),
  weights = NULL,
  level = 0.95,
  ...
)
```

Arguments

x	a bayes_ph, l0_ph or ph_fit object.
type	the aggregation scheme. `"overall"` uses the weights carried by the ph_data object (eq. 2). `"dynamic"` averages within event time $t - g$, the event-study profile. `"group"` averages within cohort, `"calendar"` within calendar period. `"dynamic"`, `"group"` and `"calendar"` need cohort and time labels on the cells.

`weights`	optional numeric vector of length K overriding the object's weights, or a K x J matrix whose columns are J separate aggregators.
`level`	credible / confidence level.
`...`	unused.

Details

For a bayes_ph posterior the aggregator is applied to **each draw** and summarised across the chain. This is the point of the Bayesian route: the reported interval propagates uncertainty about the partition itself, not merely sampling uncertainty given a grouping.

For an l0_ph or ph_fit object the aggregate is computed from the fitted effects with the plug-in variance $w'RS^{-1}R'w$, which conditions on the selected partition being correct. Remark 3 of the paper is the caveat.

Value

A data frame with one row per reported estimand, holding the estimate, standard error or posterior standard deviation, and interval bounds.

References

Arora, P. and Wagle, R. (2026). Section 3.3 and Remark 3. See `citation("phdid")` for the full reference.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)),
             cells = c("2:2", "2:3", "3:3", "3:4"))
fit <- bayes_ph(d, alpha = 1, iters = 400, burn = 100, seed = 1)
aggregate(fit, "overall")
aggregate(fit, "dynamic")
```

---

alpha_for_groups *Concentration parameter giving a target prior number of groups*

Description

Inverts eq. (23), $E[m \mid \alpha, K] \approx \alpha \log(1 + K/\alpha)$, numerically. Useful for setting a prior that is neutral between pooling and the flexible fit: `alpha_for_groups(K, K / 2)` reproduces the choice made in the paper's simulations.

Usage

```
alpha_for_groups(K, target)

expected_groups(K, alpha)
```

Arguments

K	the number of cohort-time cells.
target	the desired prior expected number of groups, between 1 and K.
alpha	the concentration parameter.

Value

the concentration parameter.

Examples

```
alpha_for_groups(18, 9)   # the paper's simulation setting, alpha ~ 7
expected_groups(7, 1)     # E[m] at the applications' alpha = 1
```

---

alpha_sensitivity *Regularisation paths: how the estimates move with the tuning parameter*

Description

Both estimators have a single knob that slides them between full pooling and the fully flexible fit — the Dirichlet Process concentration `alpha` for `bayes_ph()`, and the ℓ_0 penalty `lambda` for `l0_ph()`. These functions trace what the knob does.

Usage

```
alpha_sensitivity(
  object,
  alpha_grid = c(0.1, 0.25, 0.5, 1, 2, 5, 10, 25, 50, 100),
  type = "overall",
  level = 0.95,
  ...,
  seed = NULL
)

lambda_sensitivity(
  object,
  lambda_grid = NULL,
  type = "overall",
  by = c("lambda", "m"),
  level = 0.95,
  n_bic = NULL,
  ...
)
```

Arguments

object	a ph_data object.
alpha_grid	concentration values to trace.
type	"overall" (default), "cells" for one row per cell, or any aggregation accepted by aggregate.bayes_ph().
level	credible / confidence level.
...	further arguments to bayes_ph(), such as iters and burn.
seed	optional integer for reproducibility.
lambda_grid	penalty values to trace. NULL derives a grid from the agglomeration path so that every partition the penalty can reach is visited.
by	index the ℓ_0 path by the penalty ("lambda", the default) or by the number of groups ("m"). See the note below on why these are not the same path.
n_bic	passed to l0_ph().

Details

Reporting the path rather than a single value is the paper's own recommendation, and for a good reason. In its simulation the path is nearly flat, so the choice is immaterial; but in its seven-cell application the overall effect slides from about -0.017 under heavy pooling to about -0.038 as the model approaches the flexible fit. Choosing the tuning parameter from the data would turn it into a data-dependent object, with the usual empirical-Bayes consequences of understated posterior uncertainty and a double use of the data, so the honest report is the whole path.

lambda_sensitivity() is the frequentist counterpart. For each penalty it refits l0_ph() under the Appendix B stopping rule and records the grouped effects. Because the ℓ_0 path is a step function — the selected partition changes only when lambda crosses a merge threshold — the default grid is derived from the agglomeration path itself, so that every distinct partition on it is visited exactly once rather than being missed between grid points.

Its intervals are the plug-in intervals of l0_ph() and condition on the selected partition being correct, so they narrow as the penalty pools more cells even where that pooling is wrong. Remark 3 of the paper is the caveat; the Bayesian path does not have this problem because it averages over partitions.

Value

A data frame with one row per tuning value (and per cell when type = "cells"), holding the estimate and its interval bounds. The anchors attribute carries the fully pooled and fully flexible reference values, and param names the tuning parameter. Plot with plot_sensitivity().

Aggregate or per-cell

With type = "overall" (the default) each row is one value of the tuning parameter and the reported estimand is the overall ATT. "dynamic", "group" and "calendar" behave the same way for the corresponding aggregations.

With type = "cells" the path is reported **for every cohort-time effect separately**, one row per (tuning value, cell). This is usually the more informative view: the aggregate is famously robust to

over-pooling, so a flat overall path can hide cells that move a great deal. Watching the individual CATTs collapse toward one another as the penalty tightens shows exactly which cells the grouping is merging, and at what point.

Some group counts are unreachable by any lambda

Indexing by `lambda` and indexing by `m` are genuinely different paths, and the first can skip values the second visits. The Appendix B stopping rule merges while a pair's cost falls below $\lambda|A||B|$, and the factor $|A||B|$ grows as groups absorb one another. The effective per-merge threshold $\mathrm{cost}/(|A||B|)$ is therefore not monotone in the merge order, so a later merge can be cheaper in threshold terms than an earlier one — and when that happens, raising `lambda` past the earlier merge triggers both at once and no value of `lambda` selects the partition in between.

On the paper's first application this is not hypothetical: the thresholds run 0.079, 0.686, 0.397, 1.350, 2.765, 3.571, so nothing between the second and third selects five groups. Use `by = "m"` to walk the agglomeration path one merge at a time, which visits every group count from `K` down to 1.

See Also

`plot_sensitivity()`.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))

# The overall ATT across the concentration parameter.
alpha_sensitivity(d, alpha_grid = c(0.5, 5), iters = 200, burn = 50,
                  seed = 1)

# The same path for every cohort-time effect separately.
alpha_sensitivity(d, alpha_grid = c(0.5, 5), type = "cells",
                  iters = 200, burn = 50, seed = 1)

# The l0 path, per cell, across the penalty.
lambda_sensitivity(d, type = "cells")
```

---

bayes_ph *The Dirichlet Process partial-homogeneity estimator*

Description

Places a Dirichlet Process mixture prior on the cohort-time effects and samples the posterior with the collapsed Gibbs sampler of Neal (2000, Algorithm 3). Because draws from a Dirichlet Process are almost surely discrete, realisations put positive probability on ties among the τ_k, and those ties *are* the partition. The prior assigns positive probability to every partition of the K cells while favouring parsimonious groupings, and never fixes their number.

Usage

```
bayes_ph(
  object,
  alpha = 1,
  mu0 = NULL,
  sigma0_sq = NULL,
  iters = 4000L,
  burn = 1000L,
  thin = 1L,
  chains = 1L,
  sigma2 = c("auto", "random", "fixed"),
  marginal = c("exact", "diagonal"),
  init = "dispersed",
  a0 = 0.001,
  b0 = 0.001,
  seed = NULL,
  progress = interactive()
)
```

Arguments

`object`	a ph_data object.
`alpha`	the Dirichlet Process concentration parameter.
`mu0, sigma0_sq`	mean and variance of the Gaussian base measure. `NULL` uses the data-scaled defaults described above.
`iters, burn, thin`	sweeps per chain, burn-in to discard, and thinning interval.
`chains`	number of chains. More than one enables the convergence diagnostics of `ph_rhat()`; chains start from dispersed partitions.
`sigma2`	`"auto"` (default) draws the error variance each sweep when the object carries a micro panel and conditions on the supplied covariance otherwise. `"random"` and `"fixed"` force the choice; asking for `"random"` without a micro panel is not possible and falls back with a message.
`marginal`	`"exact"` (default) evaluates the full collapsed marginal likelihood, eq. (54), at every cluster-assignment move, so the moves see the cross-cell covariance. `"diagonal"` uses the fast conjugate normal-normal shortcut that ignores it. Appendix E reports that the shortcut reproduces the same aggregates but can misstate individual co-clustering probabilities by up to 0.39, so prefer the default unless you are deliberately reproducing the shortcut.
`init`	starting partition: `"dispersed"` (varies by chain), `"singletons"`, `"pooled"`, `"random"`, or an explicit label vector.
`a0, b0`	shape and rate of the inverse-gamma prior on σ^2, eq. (28). Used only when `sigma2 = "random"`.
`seed`	optional integer for reproducibility.
`progress`	print a progress bar.

Details

This is the estimator the paper recommends for inference. Unlike `l0_ph()`, which commits to one partition and reports intervals conditional on it, the posterior here averages over partitions, so the reported uncertainty includes uncertainty about the grouping itself. In the paper's simulations that distinction is decisive: where the partition is uncertain, the ℓ_0 plug-in intervals collapse to 0.57–0.62 coverage while these credible intervals degrade gracefully to 0.79–0.81, and in the well-separated regime they attain near-nominal 0.93–0.94.

Value

An object of class `bayes_ph` with components `tau` (the posterior mean cohort-time effects, eq. 38), `lower` and `upper` (2.5th and 97.5th posterior percentiles), `coclust` (the K x K posterior co-clustering matrix of eq. 39), `m_mean` (the posterior expected number of groups), and `draws` (the retained draws, for `aggregate.bayes_ph()` and the diagnostics).

Priors

The concentration `alpha` controls the prior expected number of groups through eq. (23), $E[m] \approx \alpha \log(1+K/\alpha)$; small `alpha` favours pooling, large `alpha` favours the flexible fit. Use `alpha_for_groups()` to solve that relation for a target. The paper's simulations use `alpha = 7` at $K = 18$ (so $E[m] \approx K/2$) and its applications use `alpha = 1`.

The base measure is $G_0 = N(\mu_0, \sigma_0^2)$. The defaults are data-scaled, `mu0 = median(tau)` and `sigma0_sq = (10 * sd(tau))^2`, matching the paper's applications and keeping the prior diffuse on whatever scale the outcome happens to have. Note that a data-scaled base measure is mildly empirical-Bayes; supply fixed values if you want a prior that is genuinely independent of the data.

How much the answer depends on alpha

Not much for aggregates, and potentially a lot for the grouping when K is small. The paper's solution paths are nearly flat in `alpha` for the overall ATT in the simulation, but its first application, with only seven cells, sees the overall effect slide from about -0.017 to -0.038 across `alpha`. Rather than fix a value, report the path: see `alpha_sensitivity()`.

On the error variance

When the object came from a micro panel, σ^2 is a real parameter and `sigma2 = "random"` draws it from its inverse-gamma full conditional each sweep, which is the paper's primary specification. When the object came from a first-stage estimator, the covariance $\hat{\Sigma}$ is a fixed plug-in and there is no scalar variance left to sample, so the sampler conditions on it (Appendix C). Appendix D shows the two treatments give the same coverage and interval length to two decimals, so nothing hinges on this.

References

Arora, P. and Wagle, R. (2026). Section 3. See `citation("phdid")` for the full reference.

Neal, R. M. (2000). Markov Chain Sampling Methods for Dirichlet Process Mixture Models. *JCGS* 9(2), 249–265.

See Also

aggregate.bayes_ph() for partition-aware aggregate estimands, coclustering() for the grouping structure, alpha_sensitivity() for the prior path.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
fit <- bayes_ph(d, alpha = 1, iters = 400, burn = 100, seed = 1)
fit

# The co-clustering matrix is the honest summary of the grouping: cells 1
# and 2 recur together, and so do 3 and 4.
round(coclustering(fit), 2)
```

---

coclustering *Posterior co-clustering probabilities*

Description

Extracts the $K \times K$ posterior similarity matrix $\hat{\Pi} = [\hat{\pi}_{jk}]$ of eq. (39), the probability that cells j and k are placed in the same group.

Usage

```
coclustering(x)
```

Arguments

x	a bayes_ph object.

Details

This is the honest summary of the grouping structure. A single partition, such as the one l0_ph() returns, states that certain cells are equal; the co-clustering matrix says how firmly the data support each such statement. A diffuse matrix is not a failure of the method: with few, correlated cells it is the correct report that the fine grouping is genuinely uncertain.

Value

a K x K matrix with cell labels as dimnames.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
fit <- bayes_ph(d, alpha = 1, iters = 500, burn = 100, seed = 1)
round(coclustering(fit), 2)
```

covariance_check *Compare exact and diagonal treatment of the covariance*

Description

Runs the sampler twice, once with the exact collapsed marginal likelihood in the cluster-assignment moves and once with the diagonal (independence) shortcut, and reports how far apart they land. Appendix E finds the two agree on aggregates but that the shortcut can misstate individual co-clustering probabilities by up to 0.39 in a correlated design.

Usage

```
covariance_check(object, ..., seed = NULL)
```

Arguments

`object`	a ph_data object.
`...`	further arguments to `bayes_ph()`.
`seed`	optional integer for reproducibility.

Details

Worth running once on any new design: it tells you whether the shortcut, which is much faster, is safe for the quantities you intend to report.

Value

A list with the two fits and a summary of the discrepancies.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
cmp <- covariance_check(d, iters = 500, burn = 100, seed = 1)
cmp$summary
```

enumerate_partitions *Exact partition posterior by enumeration*

Description

When K is small enough, the posterior over partitions can be computed in closed form rather than sampled: evaluate the collapsed marginal likelihood and the Chinese Restaurant Process prior at every one of the B_K partitions and normalise. This is the benchmark of Appendix E, where the paper's first application has $K = 7$ and hence $B_7 = 877$ partitions, and the sampler is shown to agree with the exact answer to within Monte Carlo error.

Usage

```
enumerate_partitions(
  object,
  alpha = 1,
  mu0 = NULL,
  sigma0_sq = NULL,
  max_cells = 11L
)
```

Arguments

`object`	a ph_data object.
`alpha`, `mu0`, `sigma0_sq`	prior settings, matching `bayes_ph()`.
`max_cells`	refuse to enumerate beyond this many cells. Raise it deliberately; B_{12} is over four million.

Details

Use it for two things: to validate a sampler run on a small design, and, on such designs, simply to report the exact posterior instead of an approximate one. The Bell numbers grow faster than exponentially, so this is infeasible beyond about eleven cells.

Value

An object of class `ph_enumeration` with the exact posterior mean effects, credible intervals, co-clustering matrix, expected number of groups, and the full table of partitions with their posterior probabilities.

References

Arora, P. and Wagle, R. (2026). Appendix E and eq. (25). See `citation("phdid")` for the full reference.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
ex <- enumerate_partitions(d, alpha = 1)
ex

# The sampler should agree with this.
fit <- bayes_ph(d, alpha = 1, iters = 500, burn = 100, seed = 1)
max(abs(coclustering(fit) - ex$coclust))
```

`flex_twfe` *The fully flexible and fully pooled benchmarks*

Description

The two corners of the partition problem, provided so that every reported comparison runs through the same code path as the estimators being compared.

Usage

```
flex_twfe(object)

pooled_twfe(object)
```

Arguments

`object`	a ph_data object.

Details

`flex_twfe()` estimates every cohort-time effect as its own parameter, the partition into K singletons. It is unbiased but, when some effects are in fact equal, wastes $K - m$ degrees of freedom and reports unnecessarily wide intervals. It returns the first-stage estimates and covariance unchanged, which makes it a useful identity check.

`pooled_twfe()` imposes a single common coefficient on all K cells, eq. (6). Because the grouped regressor is the sum of the clean cohort-time dummies, $\tilde{D}_{\text{pool}} = \sum_k \tilde{D}_k$, this is the pooled TWFE coefficient. It is the most precise estimator available and is biased for every individual effect whenever the effects genuinely differ, by eq. (18); its intervals are short but centred on the wrong estimand, which is why the paper's coverage table reports 0.04 for it.

Value

A ph_fit object.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
flex_twfe(d)$tau     # unchanged first-stage estimates
pooled_twfe(d)$tau   # one number, repeated
```

`homogeneity_test` *Is there heterogeneity to recover?*

Description

Ask this before grouping anything. Both `l0_ph()` and `bayes_ph()` will happily return a partition of pure noise: the paper's second application finds the ℓ_0 estimator splitting 138 event effects into five "bands" and the DP estimator reporting between 2.7 and 7.6 groups, when a randomization test cannot reject a single common effect. At a signal-to-noise ratio below one, such groups are quantiles of noise, not effect clusters. This function is the guard against reading them as structure.

Usage

```
homogeneity_test(object, pre = NULL, nsim = 10000L, center = TRUE, seed = NULL)
```

Arguments

`object`	a ph_data object.
`pre`	optional numeric vector of length K holding a pre-treatment (placebo) summary for each cell, constructed symmetrically with the post-treatment estimates.
`nsim`	number of randomization draws.
`center`	centre each series before exchanging; see above.
`seed`	optional integer for reproducibility.

Details

Three assessments are reported, the first two always and the third when placebo estimates are supplied.

Value

An object of class `homogeneity_test`. Printing it gives the three assessments and a plain reading of which regime the design is in.

Provenance

The package is published alongside the paper, so it matters which parts of this function are the paper's and which are additions. Each block below is labelled, and the printed output carries the same tags.

Only assessment 3 is the paper's procedure. Assessments 1 and 2 were added for this package, principally because assessment 3 requires placebo estimates that many designs cannot supply — including the paper's own first application, whose 2004 cohort is treated from the first period of the window and so has no pre-treatment cells at all.

Why centring (package modification)

Under the null the effects share a *common* value, not a zero one, so the raw post-treatment summaries are shifted by that common effect while the pre-treatment ones are not; exchanging them unshifted injects that shift into the reference distribution and makes the test conservative. Centring removes the shift and leaves the deviations exchangeable, which is the null actually being tested. Set `center = FALSE` for the literal unshifted exchange.

In practice this matters only when the common effect is large relative to the noise. At the scale of the paper's second application the two conventions agree to three decimals, so nothing there turns on it.

References

Arora, P. and Wagle, R. (2026). Sections 5.1 and 5.2.3. See `citation("phdid")` for the full reference.

Examples

```
# Genuinely heterogeneous effects: the test rejects a common value.
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
homogeneity_test(d)

# Noise around a common value: it does not.
set.seed(1)
d0 <- ph_data(rnorm(8, 0.05, 0.02), Sigma = 0.02^2 * (0.3 + 0.7 * diag(8)))
homogeneity_test(d0)
```

`l0_ph` *The l0-penalised partial-homogeneity estimator*

Description

Recovers a partition of the cohort-time cells by minimising

$$Q(\mathcal{P}) = \mathrm{RSS}(\mathcal{P}) + \lambda\, c(\mathcal{P}),$$

where $c(\mathcal{P})$ counts the pairs of cells placed in different groups (eq. 34–35). Proposition 2 of the paper shows this is not an unrelated penalised estimator but the fixed-variance maximum a posteriori partition of the same Bayesian model that `bayes_ph()` samples from, differing only in the partition prior — a pairwise prior here, the Chinese Restaurant Process there. The ℓ_0 form is used rather than an ℓ_1 fusion penalty because it produces exact equality of coefficients rather than continuous shrinkage.

Usage

```
l0_ph(
  object,
  select = c("bic", "lambda", "m"),
  lambda = NULL,
```

```
  m = NULL,
  search = c("greedy", "exact-cost"),
  n_bic = NULL
)
```

Arguments

`object`	a ph_data object.
`select`	how to choose the point on the agglomeration path. `"bic"` (default) minimises the BIC; `"lambda"` applies the Appendix B stopping rule at the supplied `lambda`; `"m"` takes a fixed number of groups.
`lambda`	the ℓ_0 penalty, on the scale of eq. (36): a pair of singleton cells is merged when their squared difference, weighted by the harmonic mean of their effective sample sizes, falls below `lambda`. Required when `select = "lambda"`.
`m`	the number of groups, when `select = "m"`.
`search`	`"greedy"` for the paper's Appendix B algorithm, whose merge costs use the diagonal (orthogonal) approximation. `"exact-cost"` is an extension, not in the paper: it scores each candidate merge by its true GLS deviance increment under the full covariance. It is still a greedy search and so still a heuristic, but it removes the orthogonality approximation from step 1. Costs $O(K^5)$; use only for small `K`.
`n_bic`	sample size in the BIC penalty on the two-stage route. See details under `partition_bic`; the default is `K`, which reproduces the paper's reported group counts.

Value

An object of class `l0_ph`, extending ph_fit, with the additional components `path` (a data frame of deviance, RSS and BIC at every point on the agglomeration path), `partitions` (the partition at each `m`), `selected` (the chosen `m`) and `lambda`.

The two steps

Exact minimisation over partitions would require searching all B_K of them, so the estimator is computed in two steps (Appendix B):

1. **Search.** A greedy agglomerative pass starting from K singletons. At each stage the pair of groups minimising

$$\Delta\mathrm{Obj}(A, B) = \frac{n_A n_B}{n_A + n_B}(\hat{\tau}_A - \hat{\tau}_B)^2 - \lambda|A||B|$$

   is merged. This runs in $O(K^3)$ time.

2. **Re-estimation.** The grouped model is re-fitted by GLS under the selected partition, `ph_fit()`.

Step 2 is not a refinement. The merge criterion in step 1 treats the design as orthogonal, and under orthogonality the greedy group means coincide with the GLS fit (eq. 49). In a general panel they do not: cells sharing a cohort or a period leave non-zero off-diagonal entries, the greedy means fail the normal equations, and only the re-estimated fit is best linear unbiased.

Choosing the penalty

With `select = "bic"` (the default) the estimator sweeps the whole agglomeration path and picks the number of groups minimising the BIC, which needs no tuning grid. Proposition 3 reads the threshold as a rate: a correct merge costs $O_p(\sigma^2)$ while a wrong merge costs $O(N\Delta\tau^2)$, so any λ between the two separates them, and BIC's effective threshold $\lambda \asymp \sigma^2 \log(NT)$ sits in that window. Passing a numeric `lambda` instead applies the stopping rule of Appendix B directly.

What it cannot do

The recovery window narrows as the distinct effects get closer together. The paper's simulations show that at a separation of three standard errors the partition is recovered only about half the time and the selection noise erases the precision gain entirely. Run `homogeneity_test()` first: if the effects carry no recoverable heterogeneity, the groups this function returns are quantiles of noise, not effect clusters.

References

Arora, P. and Wagle, R. (2026). Propositions 2 and 3, and Appendix B. See `citation("phdid")` for the full reference.

See Also

`bayes_ph()`, which averages over partitions instead of committing to one, and `homogeneity_test()`, which asks whether to group at all.

Examples

```
# Four cells, two well-separated effect levels.
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
fit <- l0_ph(d)
fit

# The whole agglomeration path, which is what BIC is selecting over.
fit$path
```

---

ph_data *Assemble the inputs for a partial-homogeneity analysis*

Description

Every estimator in **phdid** operates on the same two objects: a vector $\hat{\tau}$ of first-stage cohort-time effects and their joint sampling covariance $\hat{\Sigma}$. This is not a convenience — it is Lemma 1 of the paper, which shows that the collapsed marginal likelihood and the group-effect posterior depend on the data only through $R'\hat{\Sigma}^{-1}\hat{\tau}$ and $R'\hat{\Sigma}^{-1}R$. Any consistent first-stage estimator may therefore be used, and nothing downstream depends on the first stage beyond that pair.

Usage

```
ph_data(x, ...)

## Default S3 method:
ph_data(
  x,
  Sigma,
  cells = NULL,
  weights = "equal",
  n_units = NA_integer_,
  check_diagonal = TRUE,
  ...
)

## S3 method for class 'MP'
ph_data(x, weights = c("cohort", "equal"), ...)

## S3 method for class 'data.frame'
ph_data(
  x,
  yname,
  idname,
  tname,
  gname,
  xformla = NULL,
  weights = c("cohort", "equal"),
  ...
)
```

Arguments

x	a numeric vector of cohort-time effects, an MP object from did::att_gt(), or a data frame holding a micro panel.
...	passed to methods.
Sigma	the K x K joint sampling covariance of x. A K-vector of standard errors is accepted as a shorthand for the diagonal covariance, but see the warning above.
cells	optional data frame or character vector labelling the K cells. If a data frame, columns g (cohort) and t (time) are used where present.
weights	the aggregation weights defining the overall ATT, eq. (2). Either "equal" (the default: each cell gets weight 1/K, as in the simulations), or a numeric vector of length K, which is normalised to sum to one.
n_units	the number of units underlying the first stage. Used only to set the default sample size in the BIC penalty; see l0_ph().
check_diagonal	warn when Sigma is diagonal. Set to FALSE when the design genuinely delivers uncorrelated cells and you do not want the reminder.

`yname`, `idname`, `tname`, `gname`
: column names holding the outcome, the unit identifier, the time period, and the cohort (period of first treatment, with `0`, `Inf` or `NA` marking the never-treated).

`xformla`
: an optional one-sided formula of covariates to partial out, e.g. `~ lpop`.

Details

`ph_data()` is generic, with three entry points:

a numeric vector `ph_data(tau, Sigma)` takes the estimates and covariance directly, for any first stage you have run yourself (**fixest**, a stacked event study, an imputation estimator, ...).

an object of class `MP` the output of `did::att_gt()`. The post-treatment cells are selected and the exact joint covariance is taken from the estimator's influence functions.

a data frame a micro panel, from which the fully flexible (interacted) TWFE model of eq. (3) is fitted internally.

From a `did::att_gt()` fit:

The `MP` method keeps the post-treatment cells ($t \geq g$) and takes the exact joint covariance from the estimator's influence functions, as $\hat{\Sigma} = V_{\text{analytical}}/n$. Because the cells share units, this covariance is dense, which is precisely what Remark 1 asks for. Fit with `bstrap = FALSE` so that `V_analytical` is populated.

The default weights are cohort sizes, so the overall ATT is the population-weighted average reported in the paper's first application.

From a micro panel:

The data frame method fits the fully flexible (interacted) TWFE model of eq. (3),

$$Y_{igt} = \alpha_i + \lambda_t + \sum_g \sum_{t \geq g} \tau_{gt} D_{igt} + \varepsilon_{it},$$

and returns $\hat{\tau}_{\text{flex}}$ together with $\hat{\sigma}^2(\tilde{D}'\tilde{D})^{-1}$. The unit and time fixed effects (and any covariates) are partialled out by alternating projections, which handles unbalanced panels; by the Frisch–Waugh–Lovell theorem the CATT estimates are the same whether covariates enter the regression directly or are residualised beforehand, which is eq. (8)–(9) of the paper.

This route assumes spherical errors, since it estimates a single $\hat{\sigma}^2$. For clustered or serially correlated errors, run a first-stage estimator that reports a robust joint covariance and pass the pair `(tau, Sigma)` to the default method instead — Lemma 1 says the downstream analysis is unchanged.

Value

An object of class `ph_data`, a list with components `tau` (the K first-stage effects), `Sigma` (their K x K covariance), `Omega` (its inverse), `cells` (a data frame of cohort and time labels), `weights` (the aggregation weights defining the overall ATT), and book-keeping used by the information criteria.

The covariance matters

$\hat{\Sigma}$ is generally **not** diagonal: cohort-time cells share units and share the unit and time fixed effects, so their estimates are correlated. Remark 1 of the paper is explicit that using the exact cross-cell covariance is what makes the reported intervals honest. Treating the cells as independent understates the posterior variance of linear aggregates such as the overall ATT — by a factor of about 3.5 in the paper's design — and misstates individual co-clustering probabilities by up to 0.39. `ph_data()` warns when it is handed a diagonal covariance so that the choice is at least deliberate.

References

Arora, P. and Wagle, R. (2026). Lemma 1 and Remark 1. See `citation("phdid")` for the full reference.

See Also

`l0_ph()` and `bayes_ph()`, which consume this object.

Examples

```
# A first stage you have run yourself: four cohort-time effects, two of
# which share a common value. The cells share units, so the covariance is
# correlated rather than diagonal.
tau <- c(0.10, 0.11, 0.42, 0.40)
Sigma <- 0.02^2 * (0.3 + 0.7 * diag(4))
d <- ph_data(tau, Sigma, cells = c("2004:2004", "2004:2005",
                                   "2006:2006", "2006:2007"))
d
```

---

ph_design *A calibrated partial-homogeneity design*

Description

Sets up the balanced staggered panel of the paper's Section 4.1: N units split equally across a never-treated group and several treated cohorts, observed over T periods. The default is the paper's headline design, $N = 2000$, $T = 10$, cohorts entering at $t = 3, 5, 7$, which yields $K = 18$ post-treatment cohort-time cells.

Usage

```
ph_design(N = 2000L, T = 10L, cohorts = c(0, 3, 5, 7), sigma = 1)
```

Arguments

`N`	number of units; must divide evenly among the cohorts.
`T`	number of periods.
`cohorts`	treatment entry periods, with `0` marking the never-treated group.
`sigma`	the error standard deviation.

Details

The design is built once and reused across replications, since the unit and time fixed effects are removed by the within transformation and only the errors vary. Everything the estimators need reduces to the $K \times K$ within cross-product $\tilde{D}'\tilde{D}$.

Value

An object of class `ph_design` carrying the cell list, the within cross-product, the effective sample sizes $n_k = \|\tilde{D}_k\|^2$, and the mean flexible standard error used to calibrate separation.

See Also

`ph_truth()` to build a partial-homogeneity parameter vector on this design, `ph_sample()` to draw one replication, and `sim_study()` to run a Monte Carlo.

Examples

```
des <- ph_design()
des
```

---

ph_fit *Fit the grouped model under a given partition*

Description

The workhorse of the package. Given a partition $\mathcal{P}$ of the K cohort-time cells with group-indicator matrix R, this solves the restricted generalised least squares problem of eq. (50) and (55),

$$\hat{\phi} = (R'\hat{\Sigma}^{-1}R)^{-1}R'\hat{\Sigma}^{-1}\hat{\tau}, \qquad \mathrm{Var}(\hat{\phi}) = (R'\hat{\Sigma}^{-1}R)^{-1},$$

and assigns each cell its group effect, $\hat{\tau}_k = \hat{\phi}_p$ for $k \in C_p$.

Usage

```
ph_fit(object, partition)
```

Arguments

`object`	a ph_data object.
`partition`	an integer vector of length K assigning each cell to a group. Labels need not be consecutive; they are canonicalised internally. A list of index vectors is also accepted.

Details

Two special cases are worth naming. The partition of K singletons returns the fully flexible estimator and its covariance unchanged; the partition with a single group returns the fully pooled estimator of eq. (6). `flex_twfe()` and `pooled_twfe()` are thin wrappers on those two calls.

Value

An object of class `ph_fit`: a list with `tau` (the K grouped cell effects), `vcov` (their K x K covariance $RS^{-1}R'$), `phi` and `phi_vcov` (the m distinct group effects), `partition`, `m`, and `deviance` (the GLS deviance under the partition, used by the information criteria).

Why this is the estimator and not the group means

The obvious shortcut — averaging the flexible estimates within each group — coincides with this fit only when the within-transformed cohort-time dummies are orthogonal (eq. 49). In a general panel, cells that share a cohort share unit fixed effects and cells in the same period share time fixed effects, so $R'\hat{\Sigma}^{-1}R$ has non-zero off-diagonal entries, the group means fail the normal equations, and they are no longer best linear unbiased. Appendix B of the paper makes this argument in full. It is also why `l0_ph()` re-estimates by GLS after its greedy search rather than reporting the merged means.

References

Arora, P. and Wagle, R. (2026). Eq. (12), (50), (55) and Appendix B. See `citation("phdid")` for the full reference.

Examples

```
tau <- c(0.10, 0.11, 0.42, 0.40)
d <- ph_data(tau, Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))

# The oracle fit, handed the true partition: cells 1-2 share an effect and
# cells 3-4 share another.
fit <- ph_fit(d, c(1, 1, 2, 2))
fit

# Pooling two cells halves the variance in a balanced orthogonal design.
sqrt(diag(fit$vcov)) / sqrt(diag(d$Sigma))
```

`ph_rhat` *Convergence diagnostics for the Gibbs sampler*

Description

Reports the Gelman–Rubin statistic $\hat{R}$ and the effective sample size for the two scalar summaries that matter: the overall aggregate effect and the number of groups. This is the check of Appendix E, where four chains from dispersed initialisations (all singletons, one pooled group, and two random partitions) return $\hat{R} = 1.00$ for both quantities.

Usage

```
ph_rhat(x, split = TRUE)
```

Arguments

x	a bayes_ph object.
split	split each chain in half before computing $\hat{R}$, which detects within-chain trends that the classic statistic can miss.

Details

$\hat{R}$ needs at least two chains, so run bayes_ph() with chains >= 2. With a single chain only the effective sample sizes are reported.

Value

An object of class ph_rhat; a data frame of diagnostics is stored in its stats component.

References

Arora, P. and Wagle, R. (2026). Appendix E. See citation("phdid") for the full reference.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
fit <- bayes_ph(d, alpha = 1, iters = 250, burn = 50, chains = 4, seed = 1)
ph_rhat(fit)
```

---

ph_sample *Draw one replication from a partial-homogeneity design*

Description

Generates within-transformed outcomes under the given true effects, fits the fully flexible model, and returns the result as a ph_data object ready for any estimator in the package. Unit and time fixed effects are removed by the transformation, so the sampling behaviour is driven entirely by the errors, as in eq. (40).

Usage

```
ph_sample(design, tau_true)
```

Arguments

design	a ph_design object.
tau_true	the true cohort-time effects, or a ph_truth() list.

Value

A ph_data object.

Examples

```
des <- ph_design(N = 300, T = 6, cohorts = c(0, 3, 5))
truth <- ph_truth(des, m_star = 2, delta = 6)
d <- ph_sample(des, truth)
l0_ph(d)$m
```

`ph_truth` *A partial-homogeneity parameter vector*

Description

Partitions the K cells into `m_star` near-equal groups and assigns each group a mean on an evenly spaced grid, with adjacent gap $\Delta = \delta \times \mathrm{sd}(\hat{\tau}_{\mathrm{flex}})$. The unit-free separation δ is the distance between adjacent group means measured in standard errors of the flexible estimates, and it is what governs whether the groups can be told apart at all.

Usage

```
ph_truth(design, m_star, delta)
```

Arguments

`design`	a ph_design object.
`m_star`	the true number of distinct effects; `1` is fully homogeneous and `design$K` fully heterogeneous.
`delta`	the separation, in flexible standard errors.

Details

The paper sweeps $m^* \in \{1, 3, 6, 9, 18\}$ and $\delta \in \{3, 6, 12\}$, with $\delta = 6$ as the headline. At $\delta = 3$ the partition is recovered only about half the time and the feasible estimators are *less* precise than flexible TWFE; at $\delta = 12$ they essentially attain the oracle.

Value

A list with `labels` (the true partition) and `tau` (the true cohort-time effects).

Examples

```
des <- ph_design()
truth <- ph_truth(des, m_star = 6, delta = 6)
table(truth$labels)
```

`plot.ph_fit` *Plot cohort-time effects grouped by the selected partition*

Description

Figure 4 (left panel) of the paper: each cohort-time effect with its interval, coloured by the group it was assigned to, with horizontal bars marking the grouped effects. Cells that share a colour and a bar were judged to share a common effect.

Usage

```
## S3 method for class 'ph_fit'
plot(
  x,
  sort = TRUE,
  level = 0.95,
 col = c("#1b9e77", "#d95f02", "#7570b3", "#e7298a", "#66a61e", "#e6ab02", "#a6761d"),
  main = NULL,
  xlab = "cohort-time cell",
  ylab = "effect",
  ...
)
```

Arguments

`x`	an l0_ph or ph_fit object.
`sort`	order cells by their flexible estimate rather than by cell label.
`level`	interval level for the flexible estimates.
`col`	a vector of group colours, recycled as needed.
`main`, `xlab`, `ylab`	plot labels.
`...`	passed to `graphics::plot()`.

Value

`x`, invisibly.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
plot(l0_ph(d))
```

`plot_coclustering` *Plot the posterior co-clustering matrix*

Description

Figure 4 (right panel): darker cells co-cluster more often. Read this rather than a single partition when the grouping is uncertain — a diffuse matrix is the correct report that the data do not pin the fine structure down.

Usage

```
plot_coclustering(
  x,
  sort = TRUE,
  main = "Posterior co-clustering probability",
  digits = NULL,
  ...
)

## S3 method for class 'bayes_ph'
plot(x, ...)
```

Arguments

`x`	a bayes_ph or `ph_enumeration` object.
`sort`	order cells by their flexible estimate.
`main`	plot title.
`digits`	if not `NULL`, overlay the probabilities rounded to this many digits.
`...`	passed to `graphics::image()`.

Value

`x`, invisibly.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
plot_coclustering(bayes_ph(d, iters = 500, burn = 100, seed = 1))
```

plot_l0_path *Plot the l0 solution path*

Description

Figure 3 (left): how the fit and the criteria move along the agglomeration path, indexed by the number of groups. Small `m` is a strong penalty. The dashed line marks the selected point.

Usage

```
plot_l0_path(x, ...)
```

Arguments

`x`	an l0_ph object.
`...`	unused.

Value

`x`, invisibly.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
plot_l0_path(l0_ph(d))
```

plot_placebo_band *Plot event effects against a placebo noise band*

Description

Figure 7: the cohort-time effects sorted, with the pooled effect as a dashed line and a shaded band showing the spread expected under a common effect. When the estimates stay inside the band there is no recoverable heterogeneity, and any partition returned by the estimators is a partition of noise.

Usage

```
plot_placebo_band(
  x,
  level = 0.95,
  main = "Cell effects against the noise band",
  xlab = "cell (sorted by effect)",
  ylab = "effect",
  ...
)
```

Arguments

x	a homogeneity_test object.
level	width of the band.
main, xlab, ylab	plot labels.
...	passed to graphics::plot().

Details

The band is the central level interval of the reference distribution: from the supplied placebo estimates when homogeneity_test() was given them, and otherwise from the first-stage covariance under the common-effect null.

Value

x, invisibly.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
plot_placebo_band(homogeneity_test(d))
```

---

plot_sensitivity *Plot a regularisation path*

Description

Draws the table returned by alpha_sensitivity() or lambda_sensitivity(). The two views answer different questions.

Usage

```
plot_sensitivity(
  x,
  term = NULL,
  band = NULL,
  label = TRUE,
 col = c("#1b9e77", "#d95f02", "#7570b3", "#e7298a", "#66a61e", "#e6ab02", "#a6761d"),
  main = NULL,
  ...
)
```

Arguments

x	a data frame from alpha_sensitivity() or lambda_sensitivity().
term	for an aggregate table with several terms, which one to plot.
band	draw the interval band. On a per-cell plot with many cells the bands overlap badly, so the default omits them there.
label	write the cell name at the right edge of each line.
col	a vector of line colours, recycled across cells.
main	plot title.
...	passed to graphics::plot().

Details

With an aggregate table (type = "overall" and friends) the top panel shows the estimate and its band against the fully pooled and fully flexible anchors, and the lower panel the number of groups. Where the path is flat the tuning parameter does not matter; where it slides, report the path rather than a single value.

With a per-cell table (type = "cells") each cohort-time effect gets its own line, drawn from its flexible value on the loose-penalty side toward the pooled value as the penalty tightens. Lines that converge are cells the method is merging, and where they meet is the penalty at which it does so. This is the view that shows what a flat aggregate path can hide: the overall ATT is robust to over-pooling, while the individual effects are not.

Value

x, invisibly.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
plot_sensitivity(lambda_sensitivity(d, type = "cells"))
```

---

plot_sim_study *Plot Monte Carlo results across the true number of effects*

Description

Figure 1: average sampling variance of the cohort-time estimates against m^*. The feasible estimators track the infeasible oracle in the partial-homogeneity region and carry a selection overhead when every cell is genuinely distinct.

Usage

```
plot_sim_study(x, y_var = "var_ratio", ...)
```

Arguments

x	a data frame of sim_study() results, stacked across m_star.
y_var	which column to plot.
...	unused.

Value

x, invisibly.

Examples

```
des <- ph_design(N = 300, T = 6, cohorts = c(0, 3, 5))
res <- do.call(rbind, lapply(c(1, 3), function(m) {
  sim_study(des, m_star = m, delta = 6, R = 3,
            methods = c("flexible", "oracle"), progress = FALSE, seed = m)
}))
plot_sim_study(res)
```

---

point_partition *A single representative partition from the posterior*

Description

The posterior mean cohort-time effects average over partitions and so match no single grouping. When one grouping must be reported, it should be chosen by minimising a loss against the whole posterior rather than by taking the most-visited partition, which is unstable. This implements the loss-based summaries the paper points to.

Usage

```
point_partition(x, loss = c("VI", "binder"))
```

Arguments

x	a bayes_ph object.
loss	"VI" (default) or "binder".

Details

Two losses are available. Binder's loss counts disagreeing pairs, weighting each pair by its co-clustering probability. The variation of information (VI) loss of Wade and Ghahramani (2018) is information-theoretic and tends to be less prone to returning too many small clusters.

Candidate partitions are the ones actually visited by the sampler, together with every cut of a hierarchical clustering of $1 - \hat{\Pi}$, which supplies sensible candidates the chain may not have visited.

Value

an integer vector of group labels, with attributes `loss` and `n_groups`.

References

Wade, S. and Ghahramani, Z. (2018). Bayesian Cluster Analysis: Point Estimation and Credible Balls. *Bayesian Analysis* 13(2), 559–626.

Lau, J. W. and Green, P. J. (2007). Bayesian Model-Based Clustering Procedures. *JCGS* 16(3), 526–558.

Examples

```
d <- ph_data(c(0.10, 0.11, 0.42, 0.40), Sigma = 0.02^2 * (0.3 + 0.7 * diag(4)))
fit <- bayes_ph(d, alpha = 1, iters = 500, burn = 100, seed = 1)
point_partition(fit)
```

---

sim_study *Monte Carlo study of the partial-homogeneity estimators*

Description

Reproduces the experiments of Section 4: for a given true number of distinct effects and separation, it draws replications and reports, for each estimator, the sampling variance of the cohort-time estimates relative to flexible TWFE, the average absolute bias, how well the true partition is recovered (adjusted Rand index), and the coverage and length of nominal intervals for both the cohort-time effects and the overall ATT.

Usage

```
sim_study(
  design,
  m_star,
  delta,
  R = 100L,
  methods = c("pooled", "flexible", "oracle", "l0", "bayes"),
  level = 0.95,
  bayes_args = list(iters = 600L, burn = 200L),
  l0_args = list(),
  seed = NULL,
  progress = interactive()
)
```

Arguments

design	a ph_design object.
m_star, delta	the truth, passed to ph_truth().
R	number of replications.
methods	which estimators to run.
level	nominal interval level.
bayes_args	a list of arguments forwarded to bayes_ph().
l0_args	a list of arguments forwarded to l0_ph().
seed	optional integer for reproducibility.
progress	print a progress bar.

Details

The oracle estimator is handed the true partition and is infeasible; it is the efficiency floor the feasible estimators are chasing.

Value

A data frame with one row per estimator.

Cost

The Bayesian estimator dominates the run time, since every replication runs a full chain. The paper uses 500 replications; the defaults here are much smaller so that an example finishes quickly. Reproducing a published row takes hours, not seconds.

References

Arora, P. and Wagle, R. (2026). Tables 2 to 5. See citation("phdid") for the full reference.

Examples

```
des <- ph_design(N = 300, T = 6, cohorts = c(0, 3, 5))
sim_study(des, m_star = 2, delta = 6, R = 4,
          bayes_args = list(iters = 150, burn = 30), seed = 1)
```

Index